

软件需求规格说明书

(Software Requirements Specification)

项目名称	无人机-无人车协同柑橘采摘系统
系统名称	控制与决策子系统
课程名称	软件系统分析与设计课程实践 / 软件工程
学 号	2327406014
姓 名	朱金涛
日 期	2025 年 11 月 14 日
版 本	V1.0

目录

1. 引言	3
1.1. 文档目的	3
1.2. 文档范围	3
1.3. 读者与阅读建议	3
1.4. 文档概述	3
2. 需求概述	4
2.1. 系统概述	4
2.1.1. 系统定位与作用	4
2.2. 功能需求分析	4
2.2.1. 核心功能需求	4
2.2.2. 通信与接口功能需求	6
2.2.3. 安全与容错功能需求	6
2.3. 非功能需求分析	6
2.3.1. 性能需求	6
2.3.2. 可靠性需求	6
2.3.3. 可维护性需求	7
2.3.4. 环境适应性需求	7
2.4. 系统集成与扩展需求	7
2.4.1. 标准兼容性	7
2.4.2. 可扩展性设计	7
2.5. 总结	7
3. 需求建模	8
3.1. 用例建模	8
3.1.1. 核心用例规约描述	8
3.1.2. 其他用例简介	12
3.2. 类对象分析	13
3.2.1. 类图	13
3.2.2. 对象图	15
4. 动态建模	16
4.1. 交互分析	16

4.1.1.	SD-001 分配采摘任务.....	16
4.1.2.	SD-004 低电量应急重分配.....	18
4.2.	状态分析	19
5.	参考文献与术语表	20
5.1.	参考文献	20
5.2.	术语表	21

1. 引言

1.1. 文档目的

本文档是《软件需求规格说明书》(SRS)，旨在详细定义“无人机-无人车协同柑橘采摘系统”之“控制与决策子系统”的功能需求、非功能需求以及行为模型。
本文档的目的是为后续的系统设计、开发实现、系统测试以及未来维护提供清晰、一致且可追溯的基准。

1.2. 文档范围

本文档定义的系统是“控制与决策子系统”。该子系统是整个协同采摘系统的核心智能控制模块，在协同系统中承担“大脑”的角色。

其范围主要包括：

- 接收并处理来自无人机和无人车的异构数据。
- 制定智能化的采摘策略与动态任务分配。
- 规划无人车的全局路径与局部避障轨迹。
- 协调无人机与无人车的时序控制，并处理系统故障与异常。

1.3. 读者与阅读建议

本文档的目标读者包括：

- **项目管理人员**：可阅读本文档以理解项目范围和功能边界。
- **系统设计与开发人员**：可依据本文档进行系统架构设计、详细设计与编码实现。
- **软件测试与质量保证人员**：可依据本文档制定测试计划和测试用例，确保系统功能符合需求。
- **课程教师与评审人员**：可阅读本文档以评估需求的完整性、一致性和可行性。

1.4. 文档概述

本文档后续章节组织如下：

- **第 2 章 需求概述**：从整体上描述系统的定位、功能需求、非功能需求以及集成扩展需求。

- **第 3 章 需求建模:** 对需求进行结构化分析, 包括用例建模 (定义系统参与者与功能) 和类对象分析 (定义系统的静态结构)。
- **第 4 章 动态建模:** 详细描述系统的行为, 包括使用时序图进行交互分析, 以及使用状态机图进行状态分析。
- **第 5 章 参考文献与附录:** 提供文档引用的参考文献和术语表。

2. 需求概述

2.1. 系统概述

控制与决策子系统是无人机-无人车协同柑橘采摘系统的核心智能控制模块, 负责整个协同作业系统的任务调度、路径规划、决策执行和安全控制。该子系统通过实时处理无人机提供的视觉信息和环境数据, 结合无人车的状态反馈, 实现智能化的采摘任务分配和执行控制。

根据农业机器人领域的发展趋势, 多机器人协同作业已成为提高农业生产效率的重要技术手段。该子系统设计遵循分布式控制架构, 采用层次化决策结构, 确保系统在复杂农业环境中的稳定运行和高效作业。

2.1.1. 系统定位与作用

控制与决策子系统在整个协同系统中承担"大脑"角色, 主要负责:

- 接收并处理无人机传输的实时图像数据和 YOLOv11 目标识别结果
- 综合分析柑橘成熟度分布信息, 制定最优采摘策略
- 规划无人车的全局路径和局部避障轨迹
- 协调无人机与无人车的任务分工和时序控制
- 监控系统整体运行状态, 处理异常情况

2.1.1.1. 技术背景与发展现状

近年来, 农业机器人技术快速发展, 多机器人协同作业系统在果园管理中的应用日益广泛。根据国际农业工程学会 (CIGR) 的研究报告, 协同机器人系统能够将果园作业效率提升 35-50%。当前主流的控制架构包括集中式控制、分布式控制和混合式控制, 其中分布式控制因其良好的扩展性和容错性被广泛采用。

2.2. 功能需求分析

2.2.1. 核心功能需求

2.2.1.1. 多源数据融合处理功能

控制与决策子系统必须具备强大的数据融合能力, 实时处理来自不同传感器的异构数据:

无人机数据处理: 系统需要接收无人机传输的高分辨率 RGB 图像数据, 处理频率不低于 10Hz。同时处理 YOLOv11 目标检测模型输出的识别结果, 包括柑橘位置坐标、成熟度评

估、置信度等信息。系统应具备数据有效性验证功能，自动过滤噪声数据和异常检测结果。

无人车状态监控：实时获取无人车的位置信息（GPS 坐标、IMU 姿态数据）、运动状态（速度、加速度）、电池电量、机械臂状态等关键参数。数据更新频率应达到 50Hz 以上，确保控制指令的实时性和准确性。

环境感知数据：整合无人车搭载的激光雷达、摄像头等设备的环境感知数据，构建实时的三维环境地图，为路径规划和避障提供基础数据支撑。

2.2.1.2. 智能决策与任务调度功能

采摘任务优先级评估：基于 YOLOv11 识别结果，系统需要建立多维度的任务优先级评估模型。评估因素包括：柑橘成熟度（基于颜色特征和形状特征）、单株果树的果实密度、采摘难度系数、距离成本等。系统应采用模糊逻辑或神经网络算法，动态计算每个采摘点的综合优先级分数。

动态任务分配策略：系统需要实现动态的任务分配算法，根据无人车当前位置、电池状态、作业进度等因素，实时调整采摘任务序列。当环境条件发生变化（如天气变化影响图像识别效果）或设备状态异常时，系统应能够快速重新分配任务，确保作业连续性。

多目标优化决策：在任务执行过程中，系统需要同时优化多个目标：最小化总体作业时间、最大化采摘收益、最小化能耗、避免设备冲突等。采用遗传算法或粒子群优化算法求解多目标优化问题，找到帕累托最优解。

2.2.1.3. 路径规划与导航控制功能

全局路径规划：系统需要建立果园的数字化地图，包括果树分布、道路网络、地形起伏、障碍物位置等信息。基于 A* 算法或 RRT 算法，规划无人车从当前位置到目标采摘点的最优全局路径。路径规划应考虑无人车的运动学约束，包括最小转弯半径、最大爬坡角度、载重限制等因素。

局部避障规划：在全局路径基础上，系统需要实现动态的局部避障功能。采用动态窗口算法（DWA）或人工势场法，实时检测移动障碍物（如其他农机设备、工作人员）并生成安全的局部避障轨迹。避障算法应保证无人车能够在保持安全距离的前提下，快速绕过障碍物并回到原规划路径。

精确定位与导航：系统需要融合 GPS、IMU、视觉里程计等多种定位技术，实现厘米级的定位精度。采用扩展卡尔曼滤波（EKF）或无迹卡尔曼滤波（UKF）算法，融合多传感器数据，提高定位的稳定性和精度。在 GPS 信号较弱的树荫区域，系统应能够依靠视觉 SLAM 技术维持定位精度。

2.2.1.4. 机械臂控制与协调功能

采摘动作规划：基于无人机提供的目标柑橘三维坐标信息，系统需要规划机械臂的采摘动作序列。采用逆运动学算法计算关节角度，生成从初始位置到采摘位置的平滑轨迹。考虑到果实的易损性，采摘动作应采用柔顺控制策略，避免对果实造成损伤。

视觉伺服控制：系统需要集成视觉伺服控制功能，利用无人车搭载的近距离摄像头，实现对目标柑橘的精确定位和抓取。采用基于图像的视觉伺服（IBVS）方法，实时调整机械臂末端执行器的位置和姿态，确保采摘精度。

碰撞检测与安全保护：在机械臂运动过程中，系统需要实时监测潜在的碰撞风险，包括与果树枝干、其他果实、无人车本体的碰撞。采用基于几何模型的碰撞检测算法，结合力/力

矩传感器反馈，实现主动的碰撞避免和被动的安全保护。

2.2.2. 通信与接口功能需求

2.2.2.1. 无线通信管理

系统需要建立稳定可靠的无线通信链路，支持无人机与地面控制站、无人车之间的实时数据传输。采用冗余通信架构，主要通信方式为 2.4GHz WiFi，备用通信方式为 4G/5G 蜂窝网络。通信协议应支持数据压缩和错误检测纠正，确保在复杂电磁环境下的通信质量。

2.2.2.2. 数据接口标准化

系统需要定义标准化的数据接口格式，支持与不同厂商的无人机和无人车设备进行集成。数据接口应采用 JSON 或 Protocol Buffers 格式，包含时间戳、数据类型、载荷内容等字段。接口设计应具备良好的扩展性，便于后续功能模块的添加和升级。

2.2.3. 安全与容错功能需求

2.2.3.1. 故障检测与诊断

系统需要具备完善的故障检测机制，实时监控各子系统的运行状态。通过设置关键参数的阈值，及时发现硬件故障、通信中断、算法异常等问题。系统应建立故障知识库，采用专家系统或机器学习方法，实现智能故障诊断和预测性维护。

2.2.3.2. 应急处理与安全策略

当检测到严重故障或安全威胁时，系统应能够自动触发应急处理程序。包括：无人车紧急停车、机械臂回到安全位置、无人机返航等操作。系统应预设多种应急预案，根据故障类型和严重程度，选择相应的处理策略。

2.3. 非功能需求分析

2.3.1. 性能需求

实时性要求：控制指令的响应时间不超过 100 毫秒，确保系统能够及时响应环境变化和任务需求。数据处理延迟不超过 200 毫秒，保证决策的时效性。

吞吐量要求：系统应能够同时处理来自多个传感器的数据流，数据处理能力不低于 1GB/小时。支持同时控制至少 5 台无人车进行协同作业。

准确性要求：路径规划精度误差不超过 10 厘米，机械臂定位精度误差不超过 2 厘米。目标识别准确率不低于 95%，误识别率不超过 2%。

2.3.2. 可靠性需求

系统可用性：系统正常运行时间不低于 99%，平均故障间隔时间（MTBF）不少于 1000 小时。具备自动故障恢复能力，一般故障的恢复时间不超过 30 秒。

数据完整性：关键数据应具备备份和恢复机制，数据丢失率不超过 0.1%。通信数据应具备

校验和纠错功能，确保数据传输的可靠性。

2.3.3. 可维护性需求

模块化设计：系统应采用模块化架构，各功能模块相对独立，便于单独维护和升级。代码应具备良好的可读性和文档完整性。

远程维护支持：系统应支持远程监控和维护功能，包括参数调整、软件升级、日志查看等操作。提供标准化的维护接口和工具。

2.3.4. 环境适应性需求

工作环境：系统应能够在户外农业环境下稳定运行，工作温度范围-10℃至 50℃，相对湿度 0-95%。具备一定的防尘防水能力，防护等级不低于 IP54。

电磁兼容性：系统应满足农业机械电磁兼容性标准，在复杂电磁环境下保持正常功能。具备抗干扰能力，避免受到其他电子设备的影响。

2.4. 系统集成与扩展需求

2.4.1. 标准兼容性

系统设计应遵循国际农业机械标准 ISO 11783 (ISOBUS) 和机器人操作系统 ROS 标准，确保与其他农业设备和机器人系统的兼容性。支持标准化的通信协议和数据格式，便于系统集成和互操作。

2.4.2. 可扩展性设计

系统架构应具备良好的可扩展性，支持功能模块的动态添加和配置。预留扩展接口，便于未来集成更多类型的农业机器人或传感器设备。支持云端计算资源的弹性扩展，应对计算密集型任务的处理需求。

2.5. 总结

控制与决策子系统作为无人机-无人车协同柑橘采摘系统的核心，需要具备强大的数据处理能力、智能决策能力和精确控制能力。通过实现上述功能需求和非功能需求，该子系统将能够有效协调无人机和无人车的协同作业，显著提高柑橘采摘的自动化水平和作业效率。系统设计应充分考虑农业应用场景的特殊性，在保证功能完整性的同时，重点关注系统的可靠性、适应性和经济性。通过采用成熟的控制算法和先进的人工智能技术，构建一个技术先进、性能稳定、易于维护的智能控制系统。

3. 需求建模

3.1. 用例建模

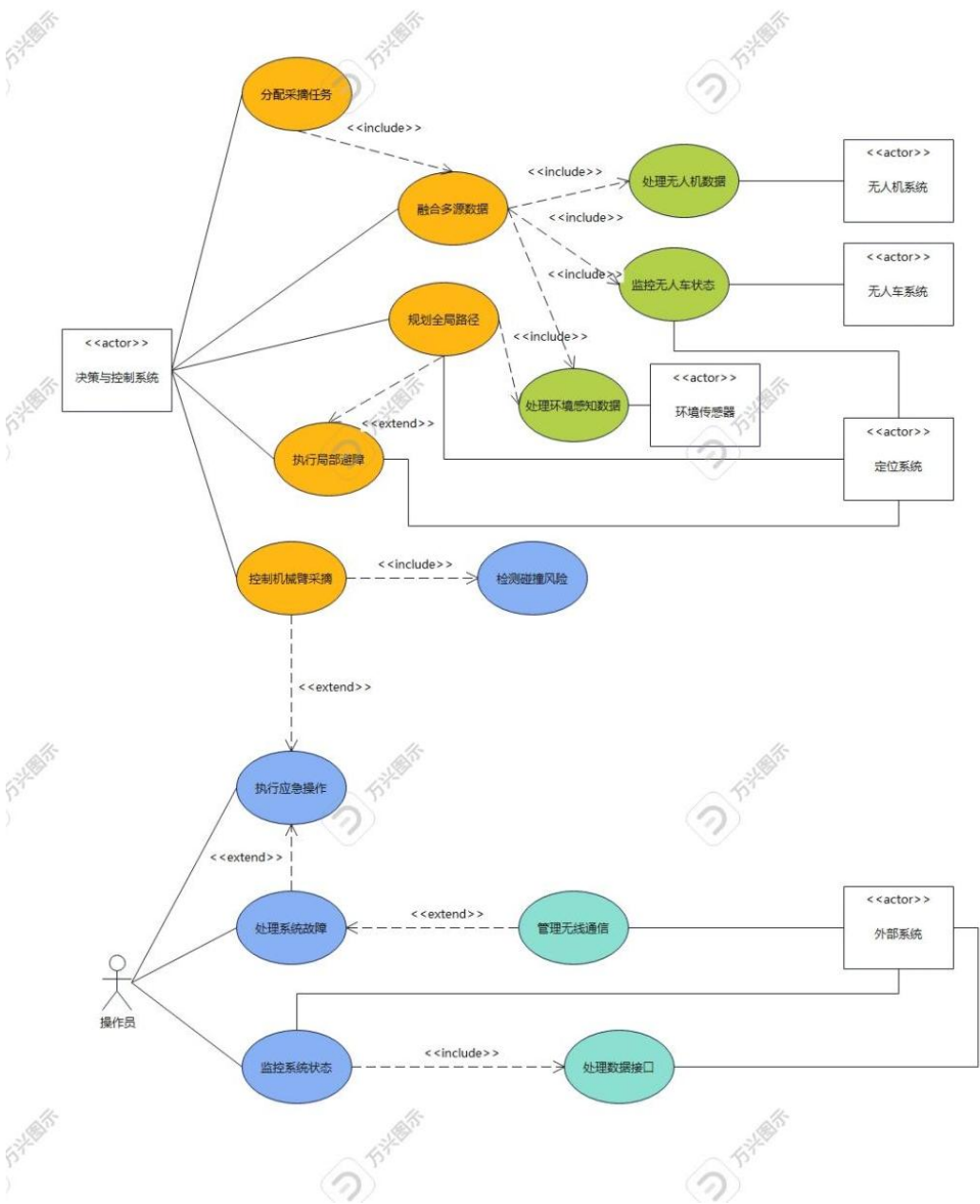


图 3-1 用例图

3.1.1. 核心用例规约描述

3.1.1.1. UC-001 分配采摘任务

用例编号	UC-001
用例名称	分配采摘任务

行为者	无人机系统、无人车系统
用例类型	核心业务用例
用例描述	系统基于无人机提供的 YOLOv11 目标识别结果，结合无人车当前状态，智能分配最优的采摘任务序列。
前置条件	无人机已完成果园扫描并传输识别结果；无人车系统处于就绪状态；控制系统正常运行
后置条件	生成优化的采摘任务列表；任务已分配给相应的无人车；系统记录任务分配结果
基本流程	<ol style="list-style-type: none"> 1. 系统接收无人机传输的柑橘识别数据 2. 分析柑橘成熟度、位置坐标、密度分布 3. 获取无人车状态信息 4. 多目标优化计算任务优先级 5. 分配任务 6. 生成任务序列并发送 7. 记录分配结果
扩展流程	<ol style="list-style-type: none"> 3a. 无人车状态异常 → 移除并重新分配 4a. 识别数据质量不佳 → 请求无人机重新扫描
异常流程	E1. 通信中断 → 使用最近有效数据临时分配并恢复通信
性能需求	任务分配计算 ≤ 5 秒；支持 5 台无人车；重分配 ≤ 3 秒

3.1.1.2. UC-002 规划全局路径

用例编号	UC-002
用例名称	规划全局路径
行为者	无人车系统、定位系统、环境传感器
用例类型	核心功能用例
用例描述	基于果园数字地图和实时环境信息，为无人车规划从当前位置到目标采摘点的最优全局路径。
前置条件	果园数字地图已加载；无人车当前位置已确定；目标采摘点坐标已知；环境感知数据可用
后置条件	生成符合约束条件的最优路径；路径已传输给无人车导航系统；路径规划结果已记录
基本流程	<ol style="list-style-type: none"> 1. 获取无人车位置 2. 提取采摘点坐标 3. 加载果园地图 4. 融合环境感知数据 5. A*算法计算路径 6. 评估可行性

	7. 选择最优路径 8. 发送路径
扩展流程	4a. 发现新障碍物 → 更新并重算 6a. 路径不满足约束 → 调整参数重算
异常流程	E1. 无可行路径 → 标记不可达并通知操作员 E2. 定位精度不足 → 切换到视觉 SLAM
性能需求	规划时间 ≤ 2 秒; 精度误差 ≤ 10 厘米; 支持实时重规划

3.1.1.3. UC-003 控制机械臂采摘

用例编号	UC-003
用例名称	控制机械臂采摘
行为者	无人车系统、环境传感器
用例类型	核心执行用例
用例描述	基于视觉识别结果, 控制机械臂执行精确的柑橘采摘动作, 包括定位、抓取、分离和收集。
前置条件	无人车已到达采摘位置; 机械臂就绪; 目标三维坐标已确定; 视觉系统正常
后置条件	成功采摘; 果实放入容器; 机械臂复位; 结果已记录
基本流程	1. 接收目标坐标 2. 精确视觉定位 3. 逆运动学规划轨迹 4. 检查碰撞 5. 视觉伺服控制 6. 抓取动作 7. 抓取成功检测 8. 分离动作 9. 放入容器 10. 复位
扩展流程	4a. 碰撞风险 → 重新规划或跳过 7a. 抓取失败 → 调整重试 8a. 分离困难 → 调整力度或切割
异常流程	E1. 定位失败 → 启用辅助照明 E2. 机械臂故障 → 停止采摘并回安全位
性能需求	采摘 ≤ 30 秒; 成功率 $\geq 90\%$; 精度 ≤ 2 厘米; 损伤率 $\leq 5\%$

3.1.1.4. UC-004 处理系统故障

用例编号	UC-004
用例名称	处理系统故障

行为者	操作员、外部系统
用例类型	安全管理用例
用例描述	实时监控系統运行状态，检测、诊断和处理各种系統故障，确保作业安全和系統稳定性。
前置条件	故障检测机制启用；预案已配置；通信正常
后置条件	故障诊断完成；处理措施已执行；信息已记录；系統恢复或安全停机
基本流程	<ol style="list-style-type: none"> 1. 监控参数 2. 检测异常 3. 自动诊断 4. 查询知识库 5. 执行处理程序 6. 评估效果 7. 记录信息 8. 通知操作员
扩展流程	3a. 未知故障 → 收集信息并通知操作员 5a. 自动处理失败 → 升级为应急处理
异常流程	E1. 严重故障 → 紧急停机、设备停止、通知操作员
性能需求	检测响应 ≤ 1 秒；恢复 ≤ 30 秒；检测准确率 ≥ 95%

3.1.1.5. UC-005 融合多源数据

用例编号	UC-005
用例名称	融合多源数据
行为者	无人机系统、无人车系统、环境传感器、定位系统
用例类型	基础数据处理用例
用例描述	实时接收、处理和融合来自不同传感器的异构数据，为上层决策提供统一、准确的数据基础。
前置条件	传感器正常；通信稳定；算法已初始化
后置条件	多源数据融合成功；生成统一环境模型；数据质量达标
基本流程	<ol style="list-style-type: none"> 1. 接收数据 2. 预处理 3. 验证有效性 4. 融合算法 5. 环境估计 6. 更新模型 7. 提供数据
扩展流程	3a. 数据异常 → 降低权重或请求重采集

异常流程	
性能需求	延迟 $\leq 200\text{ms}$; 频率 $\geq 20\text{Hz}$; 支持 ≥ 10 个数据源

3.1.2. 其他用例简介

用例名称	执行局部避障
用例描述	在全局路径基础上，实时检测和避免动态障碍物，确保无人车安全导航。

用例名称	监控系统状态
用例描述	实时监控整个协同系统的运行参数，包括设备状态、任务进度、性能指标等。

用例名称	管理无线通信
用例描述	维护与无人机、无人车的稳定通信连接，处理数据传输和协议转换。

用例名称	检测碰撞风险
用例描述	在机械臂操作过程中实时检测潜在碰撞，保护设备和作物安全。

用例名称	处理数据接口
用例描述	标准化各种数据接口格式，支持不同设备的数据交换和系统集成。

3.2. 类对象分析

3.2.1. 类图

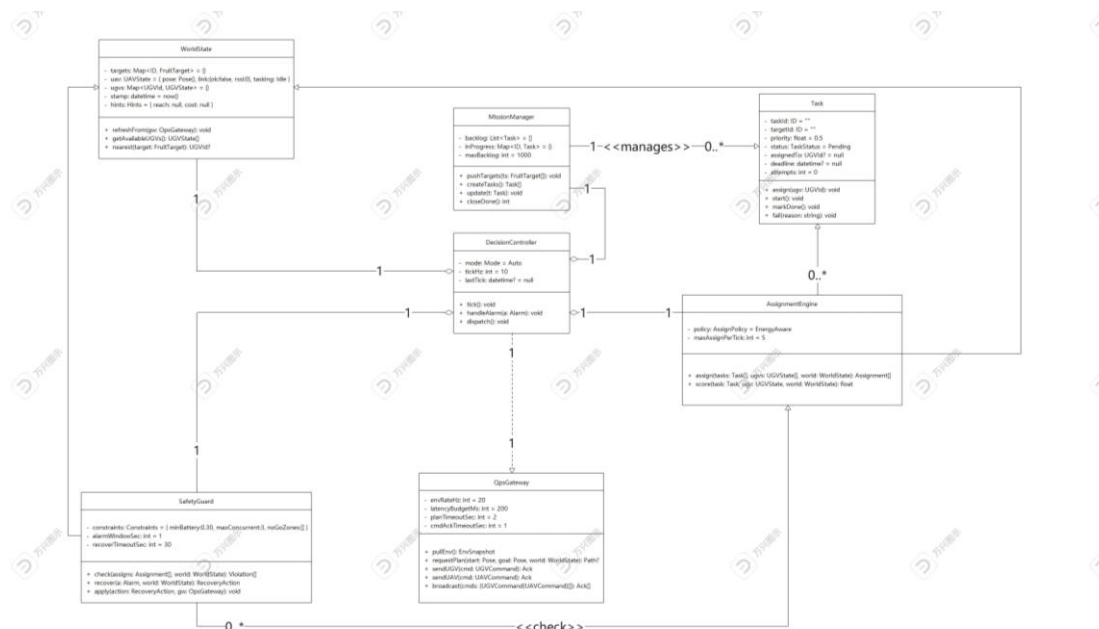


图 3-2 类图

3.2.1.1. 类图与用例图的匹配说明

在类图中，所有的类与它们之间的关系都紧密地与用例图中的功能需求一一对应。通过分析用例图，系统的核心用例被转化为类图中的具体类及其方法，确保了类图与用例图的一致性。例如：

- **任务分配 (UC-001)**：在用例图中描述了任务从接收到分配给无人车 (UGV) 的过程。类图中的 MissionManager 类负责管理任务的生成与状态更新，AssignmentEngine 负责根据策略将任务分配给 UGV，同时与 SafetyGuard 类进行约束校验，确保任务分配的合法性与合理性。Task 类作为任务实体，承载任务的具体信息（如目标、优先级、状态等）。
 - **路径规划 (UC-002)**：用例图中描述了无人车在执行任务时如何进行路径规划。在类图中，PathPlannerPort 类充当与外部路径规划服务的接口，WorldState 提供了必要的环境数据，AssignmentEngine 使用路径数据来确定任务的分配。
 - **故障处理 (UC-004)**：用例图中有描述系统的故障监测与应急恢复。在类图中，FaultMonitor 类负责检测故障并通过 SafetyGuard 生成相应的恢复动作，RecoveryManager 负责执行这些恢复操作，如紧急停车、返航或重新分配任务。
- 通过这些对比，可以清晰地看到类图中的每一个类和方法都是针对用例图中功能的实现。

3.2.1.2. 类与类之间的关系

在类图中，类与类之间的关系有明确的定义，这些关系反映了系统中各个类如何协作与交互。主要的关系包括：

聚合 (Aggregation):

- DecisionController 与 MissionManager、AssignmentEngine、WorldState、SafetyGuard、OpsGateway 是聚合关系, 表示 DecisionController 类是主要控制类, 持有并调度这些子系统的实例。
- 例如, DecisionController 通过调用 MissionManager.createTasks() 来生成任务, 通过 AssignmentEngine.assign() 来执行任务分配。SafetyGuard 在任务分配前校验任务的可行性 (如电量、负载等约束)。

依赖关系 (Dependency):

- AssignmentEngine 与 AssignmentPolicy 是依赖关系, AssignmentEngine 使用 AssignmentPolicy 来决定如何进行任务分配。
- WorldState 被 DecisionController 依赖, 用来获取当前世界的状态 (如任务、UGV、UAV 的状态等), 并为任务分配提供环境数据支持。

关联 (Association):

- MissionManager 与 Task 之间是管理关系, MissionManager 管理多个 Task 实例, 每个任务有自己的生命周期 (从创建到完成)。
- AssignmentEngine 与 Task 和 UGVState 之间有关联, AssignmentEngine 根据任务的需求和 UGV 的状态进行任务分配。

组合 (Composition):

- WorldState 通过组合关系持有 UGVState 和 UAVState 的实例, 表示 WorldState 在系统中对 UGV 和 UAV 的状态进行封装与管理。

实现 (Realization):

- AssignmentPolicy 是一个接口类, 实际的分配策略 (如 NearestFeasiblePolicy、EnergyAwarePolicy) 是其实现类, 表示 AssignmentEngine 类依赖于不同的策略来实现任务分配。

这些关系确保了系统的模块化和解耦, 每个类都有明确的职责, 通过与其他类的关系协作来完成复杂的业务功能。

3.2.1.3. 类图解释说明

类图中的每个类都承担着特定的职责, 类之间的关系反映了系统设计中的各个组件如何协作与交互。

- **DecisionController:** 作为系统的核心控制类, 它负责周期性地调度任务的生成、任务的分配、状态的校验和命令的下发。它是唯一的对外入口, 系统中的其他子系统 (如任务管理、任务分配、故障处理) 都通过 DecisionController 进行交互。
- **MissionManager:** 该类负责管理任务的生命周期, 从目标识别到任务生成, 再到任务的更新。它是任务流转的关键类, 将外部目标转换为具体的任务, 并在任务分配过程中维护任务的状态。
- **Task:** 该类表示任务实体, 存储任务的关键信息 (如目标、优先级、状态等)。Task 类不仅承载任务的基本属性, 还支持任务状态的更新 (如分配、开始、完成等)。
- **AssignmentEngine:** 该类负责根据不同的分配策略 (如最近优先、电量最优先等) 对任务进行分配, 确保任务能够合理地分配给可用的 UGV。它与 AssignmentPolicy 接口协作, 以支持多种分配策略。

- **WorldState**: 该类封装了系统中所有的环境状态数据，包括 UGV、UAV 的状态以及当前任务目标。WorldState 是 DecisionController 和其他类进行决策、任务分配的基础数据来源。
- **SafetyGuard**: 该类负责进行任务分配前的安全检查，确保所有的任务都在可行范围内（如电量足够、UGV 不超载等）。它还负责故障处理，监控系统状态，发现问题时生成恢复动作。
- **OpsGateway**: 该类是系统与外部服务交互的接口，负责拉取环境数据、请求路径规划以及下发命令到 UGV 或 UAV。它为系统提供了与实际硬件设备之间的桥梁。

3.2.2. 对象图

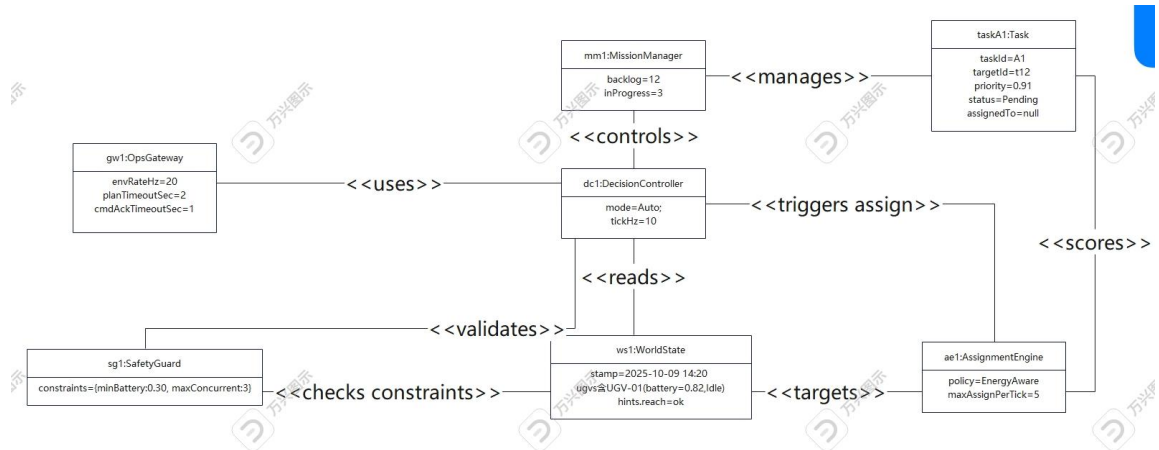


图 3-3 正常分配时刻对象图（UC-001）

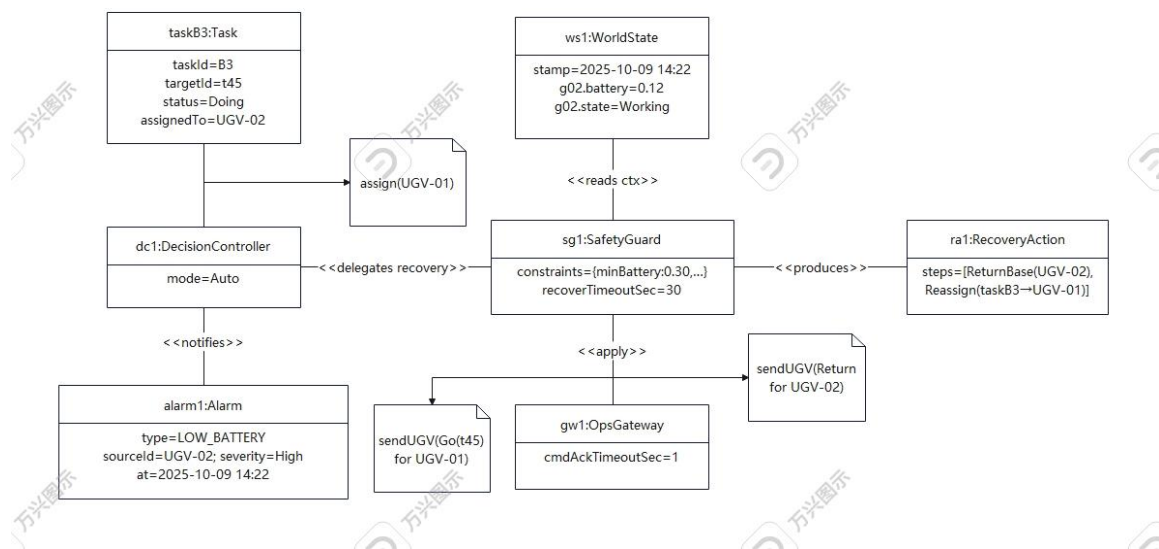


图 3-4 低电量应急重分配对象图（UC-004）

3.2.2.1. 对象图说明

对象图是类图的一个实例化，它展示了在特定时刻类对象之间的实际关系。在对象图中，我们绘制了具体的类实例，以及它们之间的交互关系。

- **正常分配时刻（UC-001）**: 对象图 A 展示了 DecisionController 调度任务管理、任务分配、约束校验和命令下发的过程。通过 Task、AssignmentEngine、SafetyGuard 和

OpsGateway，系统完成了任务的分配与状态更新。

- **低电量应急重分配 (UC-004)：**对象图 B 展示了 DecisionController 如何处理低电量告警，并委托 SafetyGuard 生成恢复计划。SafetyGuard 通过 OpsGateway 下发返航命令和重分配任务，确保系统在遇到故障时能够正常恢复。

4. 动态建模

4.1. 交互分析

4.1.1. SD-001 分配采摘任务

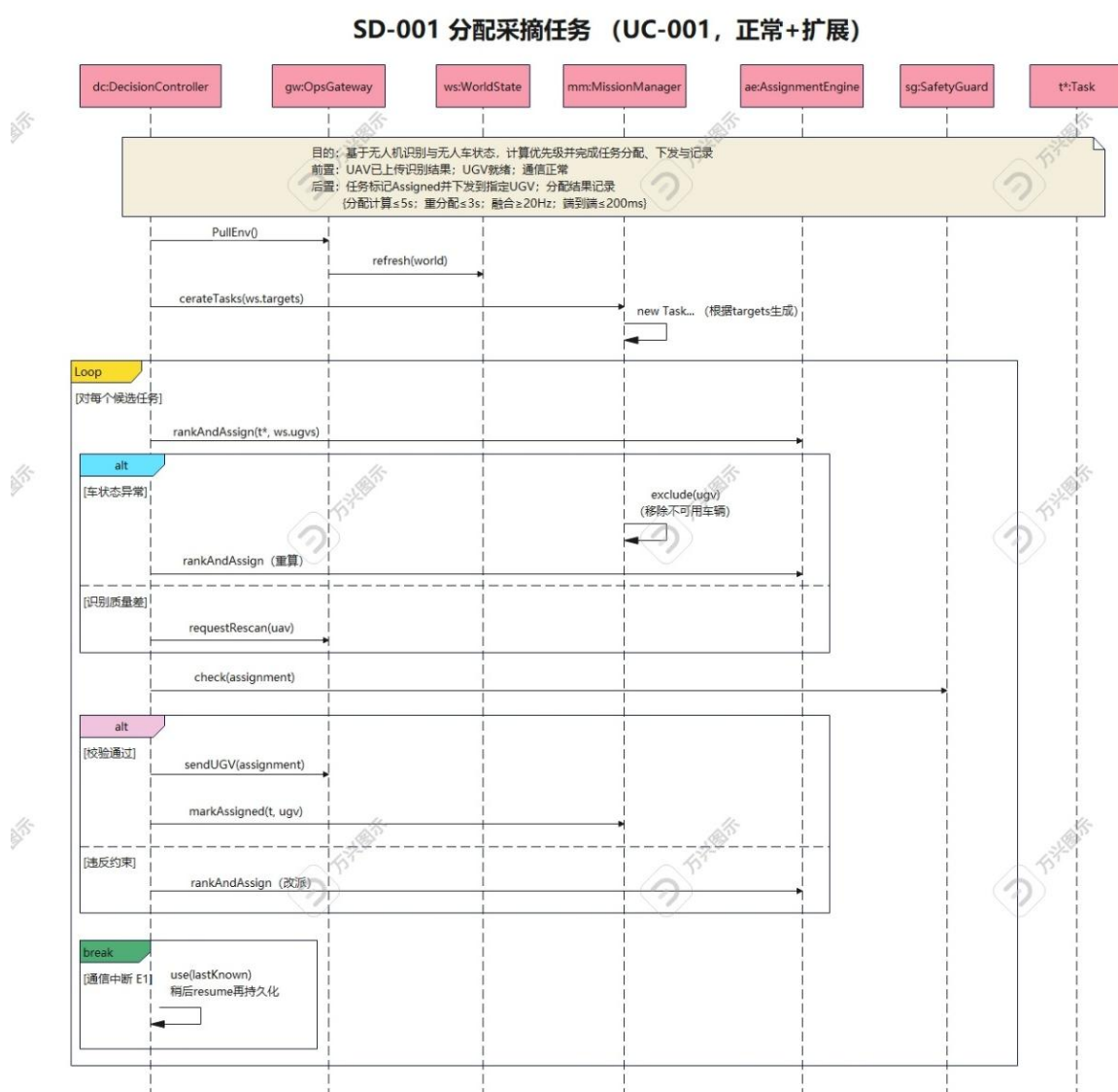


图 4-1 SD-001 分配采摘任务时序图

4.1.1.1. 规约描述

- **目的：**接收环境/识别结果，生成任务并依据车辆能力与约束完成优先级计算、任务分配与指令下发。
- **参与对象：**dc:DecisionController, gw:OpsGateway, ws:WorldState, mm:MissionManager, ae:AssignmentEngine, sg:SafetyGuard, t*:Task。
- **触发/前置：**控制循环 dc.tick() 或操作员触发；UAV/UGV 在线并可通信；世界状态可刷新。
- **主成功场景（与图中消息一一对应）**
 - S1. dc→gw: pullEnv(); gw→ws: refresh(world) 完成融合。
 - S2. dc→mm: createTasks(ws.targets), mm 依据目标生成 Task。
 - S3. dc→ae: rankAndAssign(t*, ws.ugvs) 计算优先级与分配方案。
 - S4. dc→sg: check(assignment) 校验安全/并发/能耗等约束。
 - S5. dc→gw: sendUGV(assignment) 下发到目标 UGV。
 - S6. dc→mm: markAssigned(t, ugv) 持久化分配结果。
- **扩展/异常（图中 loop/alt/break 已体现）**
 - E1（循环）：对每个候选任务执行 S3–S6。
 - E2（车状态异常）：mm.exclude(ugv), 回到 S3 重算。
 - E3（识别质量差）：dc→gw: requestRescan(uav), 任务降权/待重扫。
 - E4（通信中断）：break 使用 lastKnown 继续评估，通信恢复后 resume 再持久化。
 - E5（约束不满足）：sg 失败→回 S3 改派。
- **后置：**任务状态从 Pending/Queued→Assigned；派发记录与审计日志落库。
- **性能/非功能：**分配计算 ≤ 5s；重分配 ≤ 3s；数据融合 ≥ 20Hz；端到端控制延迟 ≤ 200ms；日志完整、幂等（重复下发不造成重复执行）。

4.1.1.2. 与类/对象建模的对应性说明

- **控制—服务—实体分层：**DecisionController、OpsGateway/MM/AE/SG，实体为 Task/WorldState；与类图的依赖/聚合方向一致。
- **关键属性/约束：**使用 Task.priority/status/assignedTo、UGV.capacity/battery、安全阈值（minBattery/maxConcurrent）等；与对象图中的实例名（如 dc1/mm1/ws1）和状态变化（Assigned）一致。

4.1.2. SD-004 低电量应急重分配

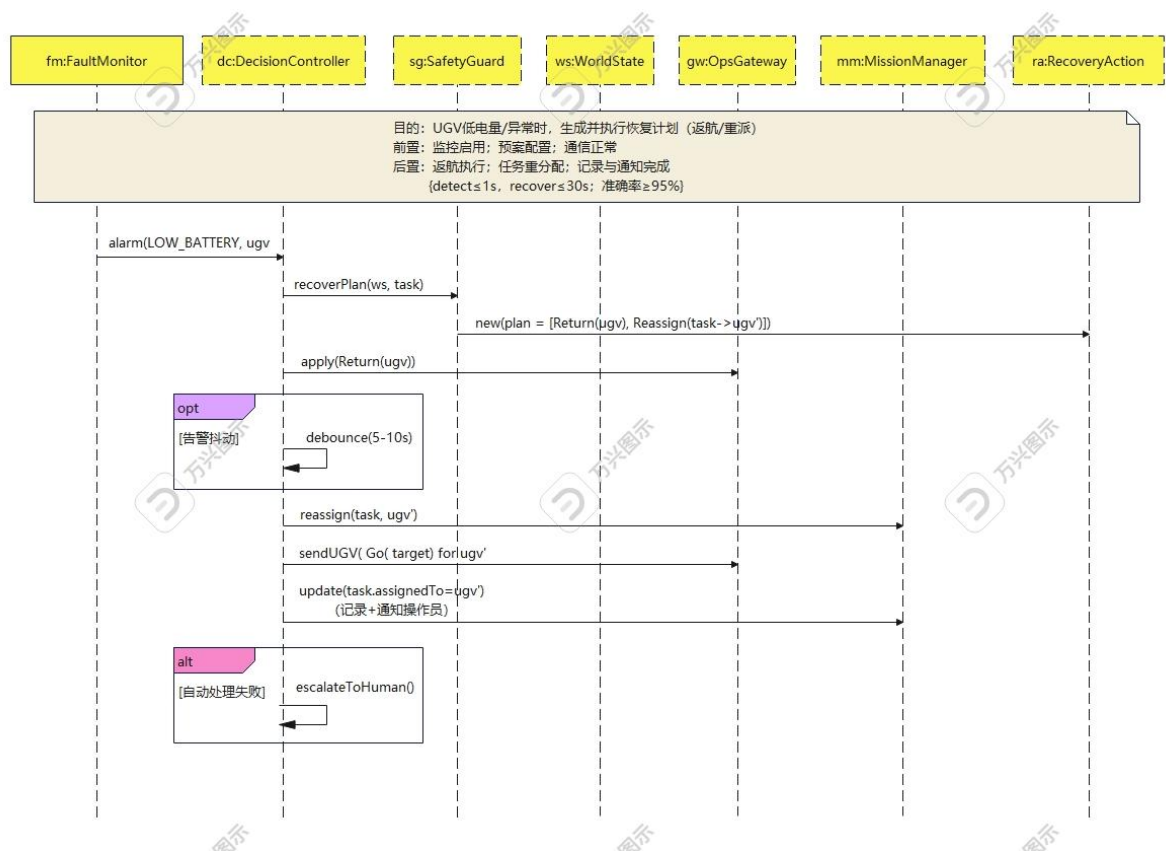


图 4-2 SD-004 低电量应急重分配

4.1.2.1. 规约描述

- **目的：**当 UGV 低电量/异常时，制定恢复计划（返航/重派/重发）并执行，保证连贯作业与安全。
- **参与对象：**fm:FaultMonitor（或告警源）、dc:DecisionController、sg:SafetyGuard、ws:WorldState、gw:OpsGateway、mm:MissionManager、ra:RecoveryAction（数据对象）。
- **触发/前置：**fm→dc: alarm(LOW_BATTERY, ugv); 监控启用、预案配置可用、链路正常。
- **主成功场景**
 - R1. dc→sg: recoverPlan(ws, task), sg→ra: new(plan) 产出步骤（如 Return/Reassign）。
 - R2. dc→gw: apply(Return(ugv)) 下发返航/安全动作。
 - R3. dc→mm: reassign(task, ugv') 选择备车; dc→gw: sendUGV(Go(target) for ugv')。
 - R4. dc→mm: update(task.assignedTo=ugv') 并通知操作员。
- **扩展/异常**

Pending (初建) → Queued (入队) → Assigned (复合: AwaitAck → AwaitReady) → Dispatched → InProgress (复合: Navigating → Working) → 终态 Completed / Failed / Canceled; 支路 OnHold、Reassigning。

- **关键事件/触发:** rankOK, assign(ugv), ack, ready/arrived, harvestDone, batteryLow, pathUnreachable, timeout, collision, cancel。
- **守卫/动作示例:**
 - Assigned.AwaitAck → AwaitReady [ack ≤ 1s] / logAssign()
 - Dispatched → InProgress: arrived
 - InProgress.Navigating → OnHold: pathUnreachable / requestRescan()
 - InProgress.Working → Completed: harvestDone
 - InProgress → Reassigning [battery < 30%] / dc.requestRecover() → Reassigning → Assigned / mm.selectNewUGV()
 - 任意 → Canceled : cancel
- **entry/exit/do:**
Pending.entry/initContext(); Queued.entry/enqueue();
Dispatched.entry/pushRoute(); InProgress.entry/startTimers()、
do/reportTelemetry()、exit/stopTimers()。
- **不变量:** Assigned ⇒ ugv.capacity ≥ task.load; InProgress ⇒ path.feasible ∧ comms.ok。
- **终结:** Completed/Failed/Canceled → [*] (确保模型有结束伪结点)。

4.2.1.2. 与类/对象建模的对应性

- 状态字段 Task.status 与类图/对象图一致, 转移由 MissionManager.assign()、OpsGateway.pushRoute()/ack、SafetyGuard.requestRescan()/recoverPlan() 等方法触发;
- 约束与属性: ackDeadline、battery、capacity、path.feasible 等与静态建模中的属性与关联一致; 对象图中 Assigned/Return 的快照能在此状态机找到对应切面。

5. 参考文献与术语表

5.1. 参考文献

- [1] Zhang, L., et al. "Multi-robot coordination for precision agriculture: A comprehensive survey." *Computers and Electronics in Agriculture*, vol. 185, 2021, pp. 106-119.
- [2] 中华人民共和国农业农村部. 《农业机器人技术发展指导意见》. 农机发〔2022〕15号, 2022.
- [3] International Commission of Agricultural and Biosystems Engineering (CIGR). "Guidelines for autonomous agricultural robots." *CIGR Journal*, vol. 24, no. 2, 2022, pp. 15-28.
- [4] Liu, S., et al. "Path planning and obstacle avoidance for autonomous agricultural robots: A review." *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 3, 2022, pp. 1825-1839.

- [5] ISO 11783:2019. "Tractors and machinery for agriculture and forestry - Serial control and communications data network." International Organization for Standardization, 2019.
- [6] Wang, H., et al. "Vision-based fruit recognition and harvesting robots in agriculture: A survey." *Biosystems Engineering*, vol. 215, 2022, pp. 261-281.
- [7] 国家标准化管理委员会. GB/T 35487-2017 《农业机器人通用技术条件》. 中国标准出版社, 2017.
- [8] Chen, Y., et al. "Real-time path planning for multi-robot systems in dynamic environments." *Robotics and Autonomous Systems*, vol. 142, 2021, pp. 103-115.

5.2. 术语表

术 语 / 缩 写	中文全称	英文全称 (如有)	描述
DWA	动态窗口算法	Dynamic Window Approach	一种用于局部避障规划的算法。
EKF	扩展卡尔曼滤波	Extended Kalman Filter	一种用于融合多传感器定位数据的算法。
GPS	全球定位系统	Global Positioning System	用于获取无人车的位置坐标信息。
IBVS	基于图像的视觉伺服	Image-Based Visual Servo	一种用于实现机械臂精确定位和抓取的控制方法。
IMU	惯性测量单元	Inertial Measurement Unit	用于获取无人车的姿态数据。
IP54	防护等级 54	Ingress Protection 54	衡量系统防尘防水能力的国际标准。
ISOBUS	国际农业机械标准 (ISO 11783)	(ISO 11783)	农业机械的串行控制和通信数据网络标准。
MTBF	平均故障间隔时间	Mean Time Between Failures	衡量系统可靠性的指标。
ROS	机器人操作系统	Robot Operating System	本系统设计遵循的机器人软件框架标准。
SD	时序图	Sequence Diagram	动态建模中用于交互分析的图表, 如 SD-001。

术 语 / 缩 写	中文全称	英文全称 (如有)	描述
SLAM	(视觉)即时定位 与地图构建	Simultaneous Localization and Mapping	用于在 GPS 信号弱时维 持定位精度的技术。
STM	状态机	State Machine	动态建模中用于状态分 析的图表，如 STM- Task。
UAV	无人机	Unmanned Aerial Vehicle	本协同系统中的空中单 元，负责传输图像和识 别结果。
UC	用例	Use Case	需求建模方法，用于描 述系统的功能单元，如 UC-001。
UGV	无人车	Unmanned Ground Vehicle	本协同系统中的地面单 元，负责执行路径规划 和采摘。
UKF	无迹卡尔曼滤波	Unscented Kalman Filter	一种用于融合多传感器 定位数据的算法。
YOLOv11	(YOLOv11) 目 标 检测模型	You Only Look Once v11	用于实时目标（柑橘） 识别的算法模型。